UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/769,535	01/30/2004	Milton E. Moskowitz	H0005134- -1623	8642

128 7590 01/15/2008

HONEYWELL INTERNATIONAL INC.
101 COLUMBIA ROAD
P O BOX 2245
MORRISTOWN, NJ 07962-2245

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
01/15/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/769,535	MOSKOWITZ ET AL.	
	Examiner	Art Unit	
	Tuan A. Vu	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 November 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-18,21 and 22* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-18,21-22* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/ are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 11/13/07.

As indicated in Applicant's response, claims 1, 7, 12 have been amended. Claims 1-18, 21-22 are pending in the office action.

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

3. Claims 1, 7, 12 are rejected under 35 U.S.C. 102(e) as being anticipated by Charisius et al, USPN: 6,983,446 (hereinafter Charisius).

As per claim 1, Charisius discloses a method for verifying a generated computer code having a plurality of lines (e.g. Fig. 4) generated from a model file of a system comprising:

processing the model file (e.g. Fig. 3; packages, class, members - col. 5, lines 50-64; *Use case Modeling* - col. 15, lines 37-44, 55-64; attribute declarations – Table 4; inheritance Metrics Table 5) to determine an expected computer code having a plurality of lines (Fig. 3-5; col. 5, lines 5-15) based on the model file (e.g. Fig. 3; Fig. 13-14 – Note: imported from UML model class definitions and OO constructs - col. 5, lines 5-15 - read on *expected code*– see col. 5, lines 61-64; col. 6, lines 12-22; Fig. 5; *definitions, templates*, Fig. 8B-C;col. 7, lines 63 to col. 8, line 21; Fig. 19A, 20A – being determined in TMM form or template of lines – SCI model Fig. 3-4);

comparing the generated computer code and the expected computer code (e.g. Fig. 4; Fig. 19B-C; Fig. 8B-C; code 1302 – Fig. 13 – Note: actual source code using the SCI reads on *generated code* lines 810, 812, 1302; and audit module to validate expected code being model specifications that encompass rules definitions – audited by metrics - reads on comparing expected and generated) to determine if the generated computer code and the expected computer code match (e.g. Figs. 8, 13, 19; Table 1-9; col. 3, lines 38-44, 55-63 – Note: Verification and audit console reads on determining by comparing whether actual code matches expectation); and

transmitting an error message (e.g. col. 3, lines 38-44, 55-63) if the generated computer code and the expected computer code do not match.

As per claim 7, Charisius discloses a computer-readable storage medium containing a set of instructions for verifying a generated computer code having a plurality of lines, the generated computer code automatically generated from a model file of a system (Fig. 13, 19), the set of instructions comprising code that:

reads in the model file (Fig. 3; col. 5, lines 50-64; *Use case Modeling* - col. 15, lines 37-44, 55-64);

determines an expected computer code (col. 5, lines 5-15) having a plurality of lines (Fig. 3; packages, class, members - col. 5, lines 50-64; *Use case Modeling* - col. 15, lines 37-44, 55-64; attribute declarations – Table 4; inheritance Metrics Table 5) based on the model file (e.g. Fig. 3; Fig. 13-14; col. 5, lines 61-64; *Use case Modeling* - col. 15, lines 37-44, 55-64; Fig. 5; *definitions, templates*, Fig. 8, col. 7, lines 63 to col. 8, line 21; Fig. 19A -Note: Note: imported from templatized model or OO definitions reads on expected code - col. 5, lines 5-15); code that reads in the generated computer code (Fig. 8A-B); and

compares the generated computer code to the expected computer code (e.g. Fig. 4; Fig. 19B-C; Fig. 8B-C; code 1302 – Fig. 13 – Note: actual source code using the SCI reads on *generated code* lines 810, 812, 1302; and audit module to validate expected code being model specifications that encompass rules definitions – audited by metrics - reads on comparing expected and generated) to determine if the generated computer code and the expected computer code match (e.g. Figs. 8, 13, 19; Table 1-9; col. 3, lines 38-44, 55-63 – Note: Verification and audit console reads on determining by comparing whether actual code matches expectation); and

transmitting an error message (e.g. col. 3, lines 38-44, 55-63) if the generated computer code and the expected computer code do not match.

As per claim 12, Charisius discloses a system for verifying the contents of a generated computer code generated from a model file (Fig. 3; col. 5, lines 50-64; *Use case Modeling* - col. 15, lines 37-44, 55-64) comprising:

a processor operable to compare the generated computer code (e.g. Fig. 4; Fig. 19B-C; Fig. 8B-C; code 1302 – Fig. 13 – Note: actual source code using the SCI reads on *generated code* lines 810, 812, 1302; and audit module to validate expected code being model specifications that encompass rules definitions – audited by metrics - reads on comparing expected and generated) with an expected computer code (e.g. Fig. 3; Fig. 13-14 .– Note: imported from templatized model definitions reads on expected code– see col. 5, lines 61-64; col. 6, lines 12-22; Fig. 5; *definitions, templates*, Fig. 8B-C; col. 7, lines 63 to col. 8, line 21; Fig. 19A, 20A – being determined in TMM form or template of lines – col. 16, lines 9-14), and

transmit an error message (e.g. col. 3, lines 38-44, 55-63) if the generated computer code and the expected computer code (e.g. col. 5, lines 5-15) do not match., the expected computer

code (e.g. Fig. 3; packages, class, members - col. 5, lines 50-64; *Use case Modeling* - col. 15, lines 37-44, 55-64; attribute declarations – Table 4; inheritance Metrics Table 5) generated by the processor based from the model file; and

a display configured to display the error message the display coupled to the processor (e.g. col. 3, lines 38-44, 55-63; *wrong* 810 – Fig. 8B).

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 2-6, 8-11, 13-18, 21-22 are rejected under 35 U.S.C. 103(a) as being unpatentable over Charisius et al, USPN: 6,983,446.

As per claims 2-4, Charisius discloses verify each of the lines of the generated computer code is in a proper format (e.g. coding styles – Fig. 19A; col. 20-31 – Note: QA/Audit tool to check source construction line by line code style against auditing rules – see Fig. 19C -- reads on proper format – see Table 11); to determine if the generated computer code includes any line of code not in the expected computer code (e.g. *synchronized ... updated automatically* – col. 5, lines 34-60; Fig. 20B; *Counts the number of code line*s -Table 1); to determine if the lines of the generated computer code are in a logical order (e.g. Table 1, Table 2, table 3, table 4, col. 10-12; Fig. 19b, 19C; col. 4 line 66 to col. 5, line 9);

But Charisius does not explicitly teach transmitting an error message if the generated code under analysis is not in the proper format, or if it includes any line not in the expected

computer code, or if it is not in a proper logical order. But based on the message error to let the developer be visually informed from the onset (e.g. col. 3, lines 38-44, 55-63; *wrong* 810 – Fig. 8B), it would have been obvious for one skill in the art at the time the invention was made to implement the source code auditing by Charisius so that when any code format, line count, line logical order as metric for auditing as set forth above does not match with the that of expected code, some error messages would be visually generated in order for the developer to effectuate proper verification of the intended target code.

As per claims 5-6, Charisius discloses comparing a header information section of the generated computer code to an expected header information section to determine if the header information section of the generated computer code matches the expected header information (e.g. *Declaration* – cols. 25-26; match a declaration, col. 37, lines 1-37- Note: generated source code or class package declaration with respect to expected declaration in OO class or Use case package – see Fig. 14-15, 22 -- in an *audit* instance reads on comparing header of a class signature declaration); and comparing a generated declared variable section of the generated computer code to an expected declared variable section of an expected computer code to determine if the generated declared variables section matches the expected declared variable section (e.g. Figs 19; Declaration Style -col. 31-35; Naming style, Performance – col. 36-39).

But Charisius does not explicitly teach transmitting an error message if the header section of generated code under analysis does not match the expected header, or if the generated declared variable section does not mach the expected declared variable section; but this error transmitting limitation has been addressed in claims 2-4 above.

As per claims 8-10, refer to claims 2-4, respectively.

As per claim 11, Charisius discloses a header information (Note: signature of a OO Class reads on a formal header declaration – see cols. 25-26) section of the generated computer code to an expected header information section to determine if the header information section of the generated computer code matches the expected header information (e.g. *Declaration* – cols. 25-26; *match a declaration*, col. 37, lines 1-37- Note: generated source code or class package declaration with respect to expected declaration in OO class or Use case package – see Fig. 14-15, 22 -- in an *audit* instance reads on comparing header of a class signature declaration).

But Charisius does not explicitly teach transmitting an error message if the header section of generated code under analysis does not match the expected header, but this error transmitting limitation has been addressed in claims 2-4 above.

As per claims 13-14, Charisius discloses wherein the results of the comparison indicates if the generated computer code has all of the content of the expected computer code (e.g. Fig. 8A-B; Fig. 20; *synchronized ... updated automatically* – col. 5, lines 34-60); wherein the results of the comparison indicates if the generated computer code has any additional content not found in the expected computer code (e.g. col. 5, lines 34-60– Note: auditing tool to match each constructs of the lines of code with format required for OO syntax construction based on template and graphical representation – see Figs 11, 19, 20 – maps with indication as to any additional content is not found – see *update view* Fig. 9; *incremental code editor* – Fig. 7).

But Charisius does not explicitly teach transmitting an error message if in the generated code all the content is not matched; or if any additional content is not found. However, this error transmitting limitation has been addressed in claims 2-4 above.

As per claims 15-17, refer to claims 2-4, respectively.

As per claim 18, refer to claim 5.

As per claims 21-22, Charisius discloses comparing the generated computer code (e.g. Fig. 4; Fig. 19B-C) to the expected computer code to determine if the generated computer code includes all of the lines (e.g. *synchronized ... updated automatically* – col. 5, lines 34-60) of the expected computer code.

But Charisius does not explicitly disclose generating an error if the generated code does not include all the lines of the expected code. However, this error transmitting limitation has been addressed in claims 2-4 above.

Response to Arguments

6. Applicant's arguments filed 11/13/07 have been fully considered but they are not persuasive. Following are the Examiner's observation in regard thereto.

35 USC § 102 Rejection:

(A) Applicants have submitted that Charisius cannot teach that 'expected computer code' and 'generated computer code' are 2 codes generated from the same model (Appl. Rmrks, pg. 8, bottom). Claims 1 and 7 specifically recite 'determine an expected computer code based on the model file'. Charisius discloses model files from standard Object-Oriented libraries or UML Use case (see Fig. 15-16) files to generate templatized definitions with textual representation of OO constructs or diagrams (see Fig 8A-B; col. 5, lines 58-64) along with the metrics information or documentation related to the model or code packages being read from (see col. 7-10). The result from the model or OO packages deriving step amounts to the amount of programming language information (including class, OO inheritance definitions, IDE typical meta-representation of OO constructs – see Fig. 4); that is, all the form which can be expected of code prior to these very

constructs being validated or audited. The QA module providing the comparing of the resulting collection of information from the determining step (i.e. expected OO code definition and language specification – see col. 5, lines 5-15) and along with the generated source code being read into the editor (Fig. 8A-B) amounts to what is claimed as 'comparing' of 'expected' against 'generated'. The claimed expected code is understood not as well-structured source code but rather as collection of OO definitions or constructs being *determined* from the model packages (e.g. Charisius: Fig. 3-5). Charisius' actual amount of code being generated is what is shown as source code lines in the editor in templatized form (see Fig. 8A-B), and the expected code is analogized as the amount of model-derived class definitions and other associated inheritance meta-information – see class constructor of Fig. 4; col. 16 line 61 to col. 17, line 7), all of which being governed by rules or metrics to be used by a QA verification module. Hence, Charisius is deemed fulfilling the comparing of 'expected' versus 'generated'. The argument is not persuasive because the language of the claim does not enforce (a) deriving then generating a first set of well-structured source code lines, and then (b) using the same model, deriving and creating another set of source code lines, then have the 2 sets compared side by side. Regarding Applicants' allegation that both 'generated computer code' and 'expected computer code' are actually generated (Appl. Rmrks pg; 9, bottom), it is noted that the claim language is far from depicting a scenario by which the 'expected code' is actually generated, and so in the same form or textual representation as the 'generated' source code, both having similarly formatted lines. Lexicographically, the qualifier 'expected' should and will not bear exact same full meaning as the qualifier 'generated' unless such dual nomenclature is deliberately redefined clearly in the Specifications; which is not the case. Indeed, the Applicants' Specifications mention about (i)

some expected constructs like graphical model input/output template or header structure; and(ii) a read-in process using a previously persisted source code (which is being read into the tool), (iii) validating the read-in source line construct (see para 0023, pg. 5; para 0028-0029, pg. 7-8) based on said model respective constructs or specifications (none of which are source code lines) using a verification module. This teaching from the Specifications if (hypothetically) read into the claim is deemed not different from Charisius' deriving model constructs, its class object and their member relationship, and using a QA module to verify whether the predefined metrics associated with the constructs are fulfilled by the corresponding parts of the source code being placed inside a templatized panel/sub-panels within the GUI auditing tool (see Fig. 8A-B). It is deemed that Charisius has fulfilled the 2 computer code as claimed; and the claim in the present state cannot preclude the class and OO member specifications (e.g Fig. 3-5) from being analogized as expected program code; nor can it preclude the source code lines being derived from the template (Figs. 8, 9, 11 and related text) from being analogized to generated source code. For lack of more distinguishing details in the claim language, Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

(B) Applicants have submitted that Charisius teaches only one computer code being generated, and in contrast, claim 1 recites a second computer code being generated and compared against a first (generated) computer code (Appl. Rmrks pg. 10, middle). The claim language has been interpreted with the analysis set forth in section A; i.e. the expected code not treated as code lines like source code being generated in an editor tool (even if the Specifications were to be read

into the claim). In other words, the claim language is not provided with sufficient format-related teachings in terms of representation or form for said first or second computer code to enforce a clear understanding of 'expected' and 'generated' plurality of lines; and this insufficiency cannot preclude Charisius (refer to section A) from reading away from 'expected' and 'generated' computer code, and proffering that Charisius' tool is for synchronizing only appears largely an incorrect statement in view of the verification endeavor by Charisius. Based on the interpretation of what the claimed 'generated'versus 'expected' computer code amounts to, and the issues mentioned in section A, the argument is by far not sufficient to successfully point out how the language of the claims patentably distinguishes them from the references. One skill in the art would not give the term 'expected' code the following very full meaning: that this is source code representation as lines thereof and formatted same as the generated source code for comparison as a side-by-side approach in a viewing tool; simply because the claim language is by far insufficient to depict such scenario.

USC 35 § 103 Rejection:

(C) Applicants have submitted that because Charisius fails to fulfill all the elements of the claim, this deficiency in Charisius is propagated to all dependent claims. Taking the analysis set forth in section A, B into consideration, it is deemed that Applicants' argument is not providing proper grounds as to point out how Charisius when combined with a secondary teaching would fail to fulfill (to render obvious) all the subject matter being addressed in each of those 103a rejected claims. The argument, in light of section A and B, will not overcome the obviousness grounds of rejection.

In all, the claims stand rejected as set forth in the Office Action.

Conclusion

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (571) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on (571)272-3756.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence - please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.
Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Tuan A Vu
Patent Examiner,
Art Unit 2193
January 13, 2008